AMENDMENTS

In the Claims:

Please amend the claims as follows:

B1

1. (Previously Presented) A computer system for efficiently executing instructions of computer programs, comprising:

processing circuitry having a pipeline, said pipeline configured to execute instructions from one of a plurality of programs, said processing circuitry configured to stop executing said one program during a first context switch in response to a first context switch command and to resume executing said one program during a second context switch in response to a second context switch command;

cache memory;

computer memory having a plurality of addresses; and

memory control circuitry coupled to said processing circuitry, said memory control circuitry, in response to said second context switch command, configured to identify one of said addresses of said computer memory that is storing a data value previously written by said pipeline during execution of an instruction of said one computer program prior to said first context switch, said memory control circuitry further configured to retrieve said data value from said computer memory in response to said second context switch command and to store said retrieved data value in said cache memory.

2. (Original) The system of claim 1, wherein said processing circuitry is further configured to execute instructions of another of said computer programs in response to said first context switch command.

3. (Original) The system of claim 1, wherein said memory control circuitry is further configured to determine, in response to said second context switch command, whether said data value was utilized by said processing circuitry to execute an instruction within a specified time period prior to said first context switch.

B1

4. (Original) The system of claim 1, wherein said memory control circuitry is configured to maintain a plurality of mappings, each of said mappings respectively correlating a data value stored in said cache memory with one of said memory addresses of said computer memory, said memory control circuitry further configured to maintain a bit of information that is associated with one of said mappings, said memory control circuitry configured to assert said bit when a data value correlated with a computer memory address via said one mapping is utilized to execute an instruction of said one program, said memory control circuitry further configured to deassert said bit periodically.

5. (Original) The system of claim 4, wherein said memory control circuitry is further configured to determine, in response to said second context switch command and based on said bit, whether said data value was recently utilized by said processing circuitry to execute an instruction prior to said first context switch.

6. (Original) The system of claim 5, wherein said memory control circuitry is further configured to store said mappings and said bit to said computer memory in response to said first context switch command and to retrieve said

mappings and said bit from said computer memory in response to said second context switch command.

7. (Previously Presented) A computer system for efficiently executing instructions of computer programs, comprising:

processing circuitry having a pipeline, said pipeline configured to execute instructions from one of a plurality of programs, said processing circuitry configured to stop executing said one program during a first context switch in response to a first context switch command and to resume executing said one program during a second context switch in response to a second context switch command;

cache memory;

computer memory having a plurality of addresses; and

memory control circuitry coupled to said processing circuitry, said memory control circuitry configured to maintain a plurality of mappings, said mappings respectively correlating data values previously written by said pipeline during execution of an instruction and stored in said cache memory with said memory addresses of said computer memory, said memory control circuitry configured to store said mappings in said computer memory in response to said first context switch command and to retrieve said data values from said addresses that are identified by said mappings stored in said computer memory in response to said second context switch command, said memory control circuitry further configured to store in said cache memory said retrieved data values.

8. (Original) The system of claim 7, wherein said processing circuitry is further configured to execute instructions of another of said computer programs in response to said first context switch command.

B1

9. (Original) The system of claim 7, wherein said memory control circuitry is further configured to maintain utilization data indicative of which of said memory addresses are storing data values accessed within a specified time period prior to said first context switch, and wherein said memory control circuitry, based on said mappings and said utilization data, is further configured to select for retrieval data values identified by one of said mappings and accessed within said specified time period, wherein each of said retrieved data values is a data value selected by said memory control circuitry based on said utilization data.

10. (Original) The system of claim 9, wherein said memory control circuitry is further configured to store said utilization data in said computer memory in response to said first context switch command and to retrieve said utilization data and said mappings in response to said second context switch command.

11. (Original) The system of claim 9, wherein said utilization data is a plurality of bits respectively associated with said mappings, wherein said memory control circuitry, for each data value accessed by said memory control circuitry, is configured to assert the bit associated with the mapping that correlates said each data value with one of said computer memory addresses, and wherein said memory control circuitry is configured to periodically deassert each of said plurality of bits.

12. (Previously Presented) A method for efficiently executing instructions of computer programs, comprising the steps of:

executing a plurality of computer programs in an interleaved fashion;

switching which of said computer programs is being executed in said executing step;

storing, prior to said switching step, at an address in computer memory a data value previously written by a pipeline in execution of an instruction corresponding to one of said computer programs in said executing step;

identifying said address in response to said switching step;

retrieving said data value from said address based on said identifying step and in response to said switching step;

storing said retrieved data value in cache memory; and

retrieving said data value from said cache memory in response to said executing step.

13. (Original) The method of claim 12, wherein said executing step further includes the step of executing instructions of a computer program in response to said switching step, and wherein said method further comprises the steps of:

determining that said address is storing a data value previously utilized in said executing step to execute an instruction of said computer program; and

performing said identifying step based on said determining step.

14. (Original) The method of claim 12, further comprising the steps of:

correlating, respectively, data values stored in said cache memory with addresses of said computer memory;

asserting a bit each time a data value correlated with said address identified in said identifying step is accessed in response to said executing step; and

periodically deasserting said bit.

15. (Original) The method of claim 14, wherein said executing step further includes the step of executing instructions of a computer program in response to said switching step, and wherein said method further comprises the steps of:

determining, based on said bit, that said address identified in said identifying step is storing a data value previously utilized in said executing step to execute an instruction of said computer program; and

performing said identifying step based on said determining step.

16. (Previously Presented) A method for efficiently executing instructions of computer programs, comprising the steps of:

executing instructions from a computer program;

halting said executing step during a first context switch in response to a first context switch command;

resuming said executing step during a second context switch in response to a second context switch command;

maintaining a plurality of mappings;

correlating, via said mappings, data values previously written by a pipeline during the executing step and stored in a cache memory with memory addresses of computer memory outside of said cache memory;

storing said mappings in said computer memory in response to said first context switch command;

retrieving, based on said mappings and in response to said second context switch command, at least one data value from at least one of said addresses identified by said mappings; and

storing said at least one retrieved data value in said cache memory.

17. (Original) The method of claim 15, further comprising the steps of:

maintaining utilization data indicative of which of said memory addresses are storing data values accessed within a specified time period prior to said first context switch; and

selecting, based on said mappings and said utilization data, data values accessed within said specified time period,

wherein said retrieving step includes the step of retrieving each data value selected in said selecting step.

18. (Original) The method of claim 17, further comprising the steps of:

storing said utilization data in said computer memory in response to said first context switch command; and

retrieving said utilization data and said mappings in response to said second context switch command.

19. (Original) The method of claim 17, wherein said utilization data is a plurality of bits respectively associated with said mappings, and wherein said method further comprises the steps of:

asserting each of said bits associated respectively with each of said mappings that identifies a data value accessed in response to said executing step; and

periodically deasserting each of said bits.

20. (Currently Amended) A computer system for efficiently executing instructions of computer programs, comprising:

a processing unit;

~~memory~~computer memory residing outside of the processing unit;

B1

cache memory; and

logic configured to store in said computer memory ~~outside of the processing unit~~ a value indicative of cache memory usage and a mapping associated with said value, said mapping indicative of a location in said computer memory storing data ~~value~~ previously requested or previously written by an instruction of a first process being executed by the processing unit when the processing unit context switches out the first process for processing of a second process, the logic further configured to retrieve ~~the~~ said data, based on said value, ~~value~~ and store said data in said cache, said processing unit ~~continue~~ continuing ~~executing~~ execution of ~~the~~ said first process with the retrieved data ~~value~~ when the processing unit context switches out the second process and context switches in the first process.

21. (Currently Amended) A method for efficiently executing instructions of computer programs, comprising the steps of:

storing in memory outside of a processing unit a value indicative of cache memory usage;

storing in said memory a mapping corresponding to said value indicative of cache memory usage, said mapping indicative of a location in computer memory storing data ~~value~~ previously requested or previously written by an instruction of a first process being executed by the processing unit when the processing unit context switches out the first process for processing of a second process;

retrieving ~~the~~ said data ~~value~~, based upon said value, when the processing unit context switches out the second process and context switches in the first process; and

continuing execution of ~~the~~ said first process with the data ~~value~~ retrieved in the retrieving step.

B1

22. (New) The system of claim 20, wherein said cache memory comprises a cache line.

23. (New) The system of claim 22, wherein said value is indicative of whether said processing unit has accessed said cache line during a particular time period.

24. (New) The system of claim 23, wherein said value indicative of said cache memory usage is defined by a flag, said logic configured to assert said flag when said first process uses said cache line.

25. (New) The method of claim 21, wherein said cache memory comprises a cache line.

26. (New) The method of claim 25, wherein said value is indicative of whether said processing unit has accessed said cache line during a particular time period.

27. (New) The method of claim 25, further comprising the step of asserting a flag when said first process uses said cache line.

28. (New) The system of claim 1, wherein said memory control circuitry is further configured to track usage frequency of cache lines in said cache memory during execution of said one program.

B1

29. (New) The system of claim 28, wherein said memory control circuitry is configured to identify said addresses of said computer memory and to retrieve said data based upon said tracked usage frequency of said cache lines.

30. (New) A system, comprising:

a processing unit;

cache memory comprising a cache line, the cache line comprising data used by a first process during execution by said processing unit; and

logic configured to track an usage frequency of said cache line, said logic further configured to store in computer memory a value indicative of the usage frequency and a mapping associated with said data used from said cache line by said first process upon a first context switch, said logic further configured to preload said data into said cache upon a second context switch based on said value.